Pitch Estimation and Comparing different Approaches

Beixian Xiong
School of Informatics and Computing
Indiana University Bloomington
Email: bxiong@iu.edu

Manish Kumar
School of Informatics and Computing
Indiana University Bloomington
Email: kumar20@iu.edu

Arpit Agarwal
School of Informatics and Computing
Indiana University Bloomington
Email: arpiagar@iu.edu

***Abstract*—As bad singers we want to know how far off we are from the right pitches of songs. In this paper, we are comparing the different approaches for pitch estimation i.e by Source separation and Sound Understanding. In Source separation approach, we first separate the source of melody from the audio signal and then apply Pitch Detection Algorithms (PDA) like YIN algorithm for pitch estimation. If we do pitch detection using YIN algorithm based on source separation results then we can estimate the pitch of singing voice in more accurate way. In second approach, we do not separate the audio signals, instead we try to understand the audio signals and estimate the pitch frequency. For sound understanding, we have used Recurrent Neural Network (RNN) and tried to make our network learn and understand the nuisance of polyphonic music signals. The results obtained by our experiments shows that Sound understanding using RNN is a very promising approach and gives better accuracy in comparison to the traditional source separation approach.**

***Index Terms*—Sound understanding, source separation, recurrent neural network, MFCC, pitch estimation**

1. Introduction

In this paper we are interested in extracting the pitch from generic audio files and compare the results obtained by different approaches to do pitch estimation. First approach includes using RNN model [2] to do source separation and then apply YIN [3] algorithm on source separated signal to estimate pitch of the signal. Second approach is based on the idea that unlike source separation method, the goal of melody extraction is to estimate the pitch of the melody and not to recover the signal of the melody source. That is, melodies can be estimated without having to separate the melody source [5]. Because of context-dependent nature of music, we decided to use RNN (Recurrent Neural Network) for the pitch estimation as it keeps a memory for the previous inputs. The baseline for both the methods would be pitch estimation using pitch detection algorithms(PDA) on the sound signal directly.

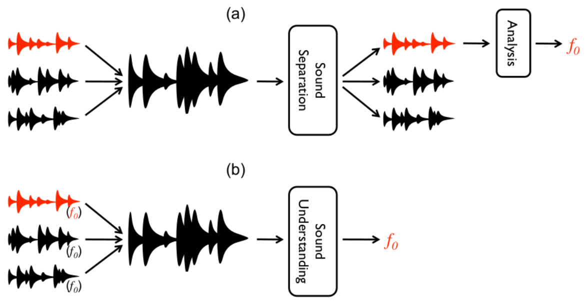

Figure 1. Diagram [7] of different approaches for melody extraction. Dig.(a) represents melody extraction using source separation. Dig. (b) represents melody extraction using sound understanding.

1.1. Definitions

We would like to make clear definitions of related concepts in this subsection.

1.1.1. Melody Extraction. Our problem is basically a melody extraction problem. To make it more precise, what we're trying to do is to detect f_0 frequencies of main melody of audio recordings, main melody being the lead singer in those recordings.

We adopt the definition of melody extraction from MIREX [1] here, which is the estimation of sequence of f0 values representing pitch of the leading voice.

1.1.2. Pitch and Frequency. In above section we treat pitch and frequency somewhat equally, but it is important to point out that pitch and frequency are the same thing only based on the fact that we're talking about the fundamental frequency. If not, actually they are two different things. Pitch is a conceptual idea while frequency is a physical attribute of sound, a pitch may have a f0 frequency, but it also contains the harmonic frequencies of that f0 frequency. Therefore all the frequencies we're focusing on will only be f0 frequency.

1.2. Related Work

Some of the traditional approaches includes salience based melody extraction and pitch-contour-based voice de-

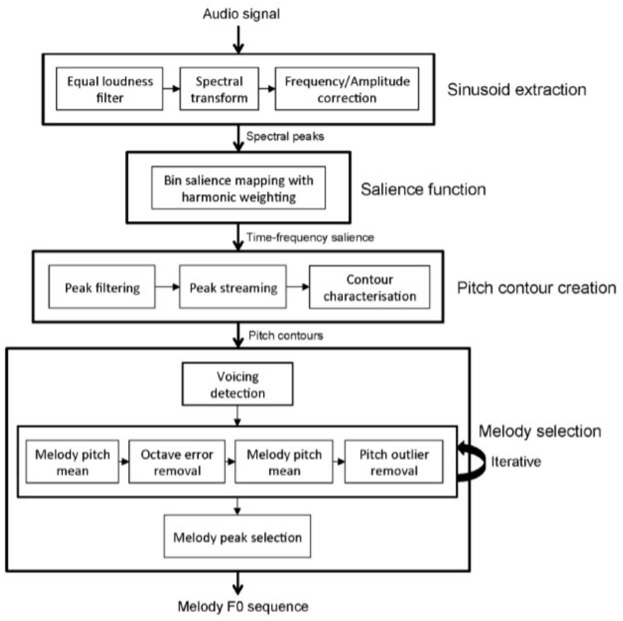

Figure 2. Block diagram [6] of the systems four main blocks: sinusoid extraction, salience function computation, pitch contour creation, and melody selection

tection [4]. State of the art approach for salience features extractions is proposed by Salmon and Gomez [6].In this paper they compute the harmonic sum of each pitch as a salience function to estimate its likelihood in an audio signal. The most salient pitches are then grouped into continuous contours, and the contours are filtered based on characteristics such as average pitch and pitch deviation. This method is divided into four main steps. First step includes filtering of signal by using Equal-Loudness Filter (ELF) before applying Short-Time Fourier Transform (STFT). This is followed by detection and smoothing of sinusoidal peaks and their frequency/amplitude values to provide more accurate estimates of the peak's true frequency and amplitude. In second step we map each extracted peak's energy to all the harmonically related f0 candidates with exponentially decaying weights to obtain salience function. In third step we do pitch contour creation which includes peak filtering , streaming and contour characterization. In the final step we do voice estimation in which the main task is to classify frames when the melody is present and when it is not. This approach works very well for vocal melodies but performs poorly on instrumental melodies.

Bosch and Gomez [4] created a variation of this method, designed to improve performance on an orchestral data set. Instead of using harmonic summing, the audio signal is modeled as a lead plus accompaniment, with the lead further approximated as a source-filter model. The pitch salience is then calculated using maximum likelihood. As expected, this method yielded better results on orchestral music, while still maintaining moderate accuracy on vocal music. Another approach would be to use deep neural network to

Rather than handcraft knowledge about musical acoustics into the system like constructing salience function based on harmonic sum, some of the researchers have proposed to use machine learning in order to train a classifier to estimate the melody note directly from the power spectrum.

Our work is similar to that of Salamons, but instead of manually specifying the salience function, we want to see if an RNN can learn to identify the notes from an audio sample. There are many interactions between sound frequencies that may be hard to model mathematically, so a neural network might be able to perform better on this task. Johnson has done related work, using a bidirectional RNN to compose music.

2. Method

In this section, we'll introduce the basic off-the-shelf pitch detection algorithm we're using as baseline, and our original approach using RNN to gain better results.

We'd like to compare whether it will help to decrease detection error to do pitch detection in just one step directly(our approach), or in two steps, that is to do source separation first then do pitch detection. The expected outcome would be that by separating out singing voice from music accompaniment, YIN algorithm should perform better in the two-step situation, but our finding proves the opposite is true, that is doing pitch detection directly will help us get best result. We also provide some possible arguments to try to explain this counter intuitive comparison result later in the discussion section.

2.1. YIN algorithm

YIN algorithm is a very accurate algorithm doing pitch detection in monophonic audio clips, according to the original paper [3] it out performs a lot of other pitch detection algorithms by only generating 1% error.

Since we're working on a pitch detection problem, we'd like to choose this classic algorithm to start with, and since we're using polyphonic audio data, it will be a good idea to apply YIN algorithm directly on our 1000 audio clips to set a performance baseline, then use our own approach to achieve better performance.

The YIN algorithm has 6 steps, each decreases the gross error to some degree.Step 1: auto-correlation method, Step 2: difference function, Step 3: cumulative mean normalized difference function, Step 4: absolute threshold, Step 5: parabolic interpolation, Step 6: best local estimate.

Steps	Gross error(%)
Step1	10.0
Step2	1.95
Step3	1.69
Step4	0.78
Step5	0.77
Step6	0.5

2.2. RNN model

Feature Representation: To be able to apply machine learning on audio data, the data must first be converted from the time to the frequency domain. There are several methods available for creating features for machine learning of audio data. For our case, we decided to use Mel-frequency cepstrum for extracting features. A significant advantage over normal spectograms is that an MFC has its frequency bands equally spaced on the mel scale, which instead of using linearly-spaced frequency bands, approximates the response of the human auditory system better. Therefore, Significant features are extracted from the songs to reduce dimensionality of the audio data and to represent them in a way that is better to process for Recurrent neural networks. The audio files are down sampled to 8KHz and then for each audio file, a spectogram of MFCCs created using the Librosa library, with a block size of 50 ms, a step size (or hop size) of 16.6 ms and 40 cepstral coefficient including the first coefficient. 80 triangular filters are used from 0 to 8 kHz. Most of these specifications for the MFCCs are taken from Schlter and Grill.

Because pitch or note of music do not occur in isolation and are context dependent, we are trying to capture the context dependency by training the network on overlapping windows (10s of milliseconds) of audio data that captures sound from before and after the current time index. For our RNN example, we use 3 time slices before and 3 after, for a total of 7 time points per window.With 40 cepstral coefficients, this is 280 data points per 16.6 ms observation.

After this transformation, all time steps of a song were combined into a single matrix and stored with the target melody frequencies to fed RNN as input.

Recurrent Neural Network: Long Short-Term Memory (LSTM) layers are a type of recurrent neural network (RNN) architecture that are useful for modeling data that has long-term sequential dependencies. They are important for time series data because they essentially remember past information at the current time point, which influences their output. This context is useful for pitch detection because of its temporal nature.

We trained our network using TensorFlow. The extracted features are input into our RNN with a signle hidden layer with two cell columns of 17 nodes and an output layer of 60 classification nodes. we call tensorflow.nn.rnncell.MultiRNNCell, which takes a list of RNNCells as its inputs and wraps them into a single layer. Each cell column within the hidden layer is split into two sections based on connectivity. The 5-node octave layer only connects to the 2000 frequency inputs, and each octave node only outputs to 12 of the 60 output notes, corresponding to an octave. The 12-node note layer is fully connected to the input and output. When activated, the output node with the maximum value is taken as the predicted melody note. To further improve our RNN network, we used layer normalization which is inspired by batch normalization.

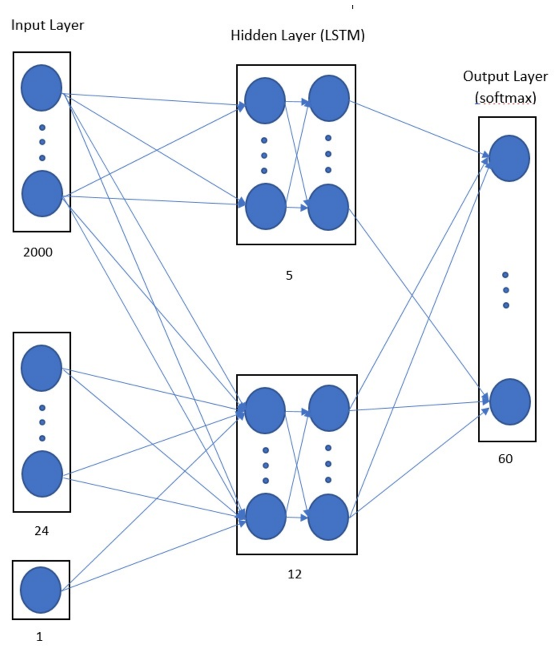

Figure 3. RNN Structure used for traning on data

Hidden layer nodes have Long Short-Term Memory (LSTM) recurrent modules. These miniature networks can store input values for near-arbitrary lengths of time. LSTM architectures vary, but all include gate nodes that control when an input will be remembered, when an input should be forgotten, and when the memorized value should be output. The frequency information in music can fluctuate wildly, so these LSTM recurrent modules provide the potential to judge when a change in inputs should be ignored and when a change is significant enough to warrant changing the note of the predicted melody. The 60 output nodes represent 60 possible output notes, and they use the softmax equation to predict the estimated pitch. Thus, The maximum value output node is taken as the output note.

The Recurrent Neural network is trained with Adam Optimizer Back-propagation to control the learning rate, which is variant of the standard back-propagation algorithm. We utilized the TensorFlow provided tf.train.AdamOptimizer for this purpose. The AdamOptimizer improves on traditional gradient descent by using momentum (moving averages of the parameters), facilitating efficient dynamic adjustment of hyper parameters. We trained the network for approx. 300 iterations.

3. Evaluation

3.1. Data

The data we're using for test and evaluation come from work by Chao-Ling Hsu and Prof. Jyh-Shing Roger Jang in Multimedia Information Retrieval lab, having 1000 Chinese song karaoke recordings, and originally designed for the research of singing voice separation, the data is generally referred as MIR-1K. The wave files have both music accompaniment and the singing voice recorded at left and right channels respectively.The ground truth files(*.pv) have manually annotated frequencies at 10-millisecond precision.The duration of each clip ranges from 4 to 13 seconds, and the total length of the dataset is 133 minutes. These clips are extracted from 110 karaoke songs which contain a mixture track and a music accompaniment track. These songs are freely selected from 5000 Chinese pop songs and sung by group of 8 females and 11 males. Most of the singers are amateur and do not have professional music training. Out of these 1000 songs, we randomly selected 750 songs for training purpose and rest of the 250 songs served as our test data. To prevent overfitting, only 600 out of 750 songs are used for training the model while 150 songs are used for validating the model.

3.2. Gross Error

We adopt the idea of gross error directly from the original paper [3] of YIN algorithm: a gross error is counted when values differ more than 20% from ground-truth frequencies. And the overall gross error of a candidate method will be the average of gross error rates of all audio clip experiments.

There's another practical reason why we choose frequency over midi representation for evaluation: in some situations singers can be off by half a note rather than a whole music note, using frequency instead of midi representation will help us detect that.Therefore we will use gross error as our evaluation metric.

3.3. Results

Algorithms	Gross Error(%)
YIN + Polyphonic Music	57.409985773
YIN + Source Separation	45.545126568
RNN + Polyphonic Music	31.3276541423

The gross error on the test set for YIN Algorithm on Polyphonic music was approx. 57% while for the same algorithm on source separated signal gross error was approx. 45%. Using RNN Algorithm, gross error on test set was approx. 31%. This relatively low error rate likely demonstrates that the recurrent neural network is capable of finding patterns in the frequency and predict the melodic pitch. From the results we can infer that the neural network is capable of generalizing some of its predictive power to music.

By doing in-depth analysis of gross error, we realized that most of the errors occurs due to the presence of Percussion Sound. Percussion Sound corresponds to the kick or drum sound which often has high power levels in the frequency range and causes sudden spikes in the frequency. For obvious reason, our network failed to model these spikes in frequency range. We feel that percussion removal in

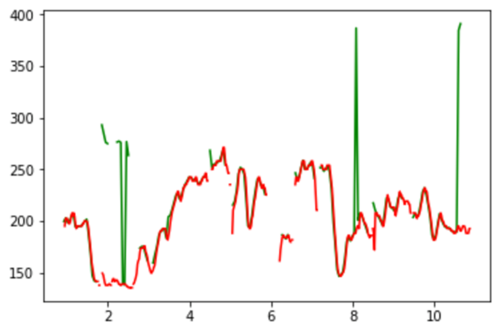

Figure 4. One of the results obtained shows how network fails to model spike caused by loud percussion.

preprocessing steps can result into significant imporvement in accuracy.

Inspite of these errors, we can conclude that RNN networks are not only able to predict the predominant tones in the melody, but also capture several nuances in the melody. This is an encouraging result in terms of future work, and may be when coupled with other methods such as removal of percussion from the music in the future can lead to groundbreaking work.

4. Conclusion and Future Work

From our results, it is quite evident that the use of harmonic data in addition to raw frequency data, fed through a recurrent neural network, can be effective at predicting the melody of harmony rich pieces. The harmonic patterns learned and recognized by our neural network represent a foundation for future work in generalizing the system to a wider range of music.

As part of the preprocessing on audio samples, we can try to remove the spikes in frequency that occur when loud percussion such as a kick or drum occurs. A kick or drum often has high power levels in the frequency ranges also shared by vocal and other instruments, and can thus obscure the melody information that we want to extract. So, it might result into better accuracy.

Because of lack of time and resources, we were not able to try additional network architecture for better accuracy. We can also try Maxpooling and drop out in our architecture. we can also try extracting features using CNN instead of using MFCC. We would also like to experiment with pitch contours [7] as features for our RNN network. So, basically in Fig. 2 after pitch contours creation, we are thinking to make use of RNN network for estimating pitch sequence. To further improve the accuracy, we can also employ post-processing smoothing such as Gaussian filter and Mean filter for classification.

We are confident enough that RNN coupled with better feature extraction methods lays the foundation for future work in the area of melody extraction without having to separate source of melody signal. It is a encouraging step towards melody extraction by sound understanding.

Acknowledgments

We would like to thank Dr. Minje Kim for his suggestions and support. Also we would like to thank the UITS at the Indiana University for providing Tensor Flow on Big Red2 supercomputers. This work is a part of project in the coursework " Machine Learning in Signal processing" taught by Dr. Minje Kim.

References

[1] J Stephen Downie. *The music information retrieval evaluation exchange(2005-2007): A window into music information retrieval research*. Acoustical Science and Technology,29(4):247-255,2008.

[2] P. S. Huang, M. Kim, M. Hasegawa-Johnson and P. Smaragdis, *Deep learning for monaural speech separation* 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, 2014, pp. 1562-1566. doi: 10.1109/ICASSP.2014.6853860

[3] Alain de Cheveign, Hideki Kawahara, *YIN, a fundamental frequency estimator for speech and music* The Journal of the Acoustical Society of America, Vol. 111, No. 4. (2002), pp. 1917-1930

[4] J. J. Bosch, R. M. Bittner, J. Salamon, and E. Gmez *A Comparison of Melody Extraction Methods Based on Source-Filter Modelling* Proc. 17th International Society for Music Information Retrieval Conference (ISMIR 2016), New York City, USA, Aug. 2016.

[5] Eric D. Scheirer *Music-Listening Systems* Doctor of Philosophy at the Massachusetts Institute of Technology, June,2000

[6] J. Salamon and E. Gomez, *Melody Extraction From Polyphonic Music Signals Using Pitch Contour Characteristics,* in IEEE Transactions on Audio, Speech, and Language Processing, vol. 20, no. 6, pp. 1759-1770, Aug. 2012. doi: 10.1109/TASL.2012.2188515

[7] J. Salamon, *Melody Extraction from Polyphonic Music Signals*, Phd thesis Universitat Pompeu Fabra, 2013

[8] Bosch, J., and Emilia Gmez. *Melody extraction by means of a source-filter model and pitch contour characterization (MIREX 2015).* 11th Music Information Retrieval Evaluation eXchange (MIREX), extended abstract, Mlaga, Spain (2015).

[9] Degani, Alessio, et al. *A Pitch Salience Function Derived from Harmonic Frequency Deviations for Polyphonic Music Analysis.* DAFx. 2014.

[10] Durrieu, Jean-Louis, et al. *Source/filter model for unsupervised main melody extraction from polyphonic audio signals.* IEEE Transactions on Audio, Speech, and Language Processing 18.3 (2010): 564-575.

[11] Durrieu, J., Bertrand David, and Gal Richard. *A musically motivated mid-level representation for pitch estimation and musical audio source separation.* IEEE Journal of Selected Topics in Signal Processing 5.6 (2011): 1180-1191.

[12] Huang, Po-Sen, et al. *Singing-voice separation from monaural recordings using robust principal component analysis.* Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on. IEEE, 2012.

[13] Maas, Andrew L., et al. *Recurrent Neural Networks for Noise Reduction in Robust ASR.* Interspeech. 2012.

[14] Dittmar, Christian, and Meinard Mller. *Reverse engineering the amen break: score-informed separation and restoration applied to drum recordings.* IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) 24.9 (2016): 1531-1543.

[15] Edirisooriya, Thaminda, Hansohl Kim, and Connie Zeng. *Melody Extraction from Generic Audio Clips.*